Acta Scientific COMPUTER SCIENCES

Volume 4 Issue 6 June 2022

Research Article

An Artificial Intelligence Based Tour Planner

Hassan Mohamed Hassan Al Ali, Mohamed Saeed Alnaqbi, Abdullah Mohamed Alkharji, Abdullah Saeed Alyamahi, Yousef Saeed AlZaabi and Yasir Hamid*

Abu Dhabi Polytechnic, United Arab Emirates

***Corresponding Author:** Yasir Hamid, Abu Dhabi Polytechnic, United Arab Emirates.

Received: December 15, 2021

Published: May 30, 2022

Abstract

The issue of hard planning for your trip or your vacation is paramount for most tourists, especially when they have some preferences that they want to do without tourism offices. Therefore, we intended to aid this issue by bringing in our smart trip advisor. It is a tourist industry application in the United Arab Emirates. The system operates based on the tourist's or user's preferences, such as religious tourism, luxury, shopping, adventures, and so on. The user will be able to create an account that will have his preferences and some of his information. With this, the user can log in to the application and start planning for the trip. The user will choose the city of arrival. After that, the application will suggest to him the main places that he may choose, depending on his preferences. The application will create a trip plan that has the shortest path and time for the user and also ask the user to choose the restaurants, cafes, hotels, and gift shops near each place that he or she planned to visit with the highest ratings. When the user finishes the trip, he will be able to share the trip with his friends, so they can have the same fun that he had, with the ability to edit it to be suitable for them. From this application, we hope that tourists have a wonderful experience in the United Arab Emirates with their preferences and the things that they most like.

Keywords: Ant Colony Algorithm; Algorithm; Artificial Intelligence; Heuristics; Tour Planner

Introduction

Everyone knows that the United Arab Emirates is one of the top countries in the world to visit. The UAE Tourism Statistics show that the number of tourists is increasing year by year from 1995 to 2021. Tourist statistics for 2019 have increased by 79.74 percent from 2018, according to www.macrotrends.net [1], which is a huge amount that highlights the role of the tourism business. The majority of tourists and visitors to the country rely on tourism corporations, which in turn supply the tourist itinerary from A to Z, reducing income for other businesses like vehicle rental and hotels. As a result, some visitors choose independent and flexible excursions that are not related to tourist groups, so they rent a vehicle and create their flexible itinerary, and the smart Vacation system was created to assist these travelers in making the most of their trip. This is an excellent journey. It is a tourism-oriented system in the United Arab Emirates. The system operates based on the tourist's or user's preferences, such as religious tourism, luxury, shopping, adventures, and so on. The system operates on an individual basis, based on the tourist's preferences. The visitor or user must first register and then log in to the system. After checking in, he must provide personal information such as his name and date of birth, and then select the sort of tourist attraction he wishes to visit, such as adventure, religious, or opulent destinations, among others. Thereafter he will choose the locations, which will then display on

the map. As a result, tourists or visitors will have greater flexibility and enjoyment based on their personal preferences.

The rest of the paper is organized as follows. Section II presents the review of the related literature. In Section III materials and methods used in the research are described. Section IV describes the research methodology, results are presented in section V, Section VI concludes the paper.

Literature Review

Ant Colony Optimization (ACO) Algorithm was created by observing the foraging behavior of real ants. It solved lots of issues that other algorithms didn't solve efficiently. The Authors of [2,3] suggested a self-adaptive approach to this algorithm to make it more efficient. Because of some downsides inherent in the design of the algorithm itself such as stagnation behavior and long search time [2]. So the proposed improvements made the algorithm nearly finish the TSP problem.

With the analogy of ants as our pointers inside the graph. Each ant must drop pheromones so other ants will likely follow it. So Stigmergy is the communication method between real ants. We followed the mechanism of local communication which means they will use certain signs for another ant to follow [4] Inside the proposed algorithm there will be a parameter that will act in place of pheromones.

Authors in [5] indicated that optimization done in every industry at their time of writing is very critical. Their paper used a variant of both the TSP and ACO, one variant was the Asymmetric Traveling Salesman Problem and the other one was a specific variant of ACS (Ant Colony System). The Difference in this algorithm from the original is that every ant is sent out sequentially and from node 1 rather than assigning them a random node in the graph and going from there.

The authors of [6] followed the same approach as the Authors of [5], Where they used the same variants of both the problem and proposed algorithm. They introduced several constraints that are specific to their problem. However, imposing a constraint on the algorithm will reduce the search space. This improves the performance of finding the near-optimal or optimal solution to a given problem. An example of a constraint is the final destination of a delivery system may be the path will change over time, But the final destination will remain the same for a given case.

Some researchers [7] improved on the original ACO algorithm and added some aspects of the known algorithm A*. When combining the evaluation method of A* with ACO we reached a more performant algorithm than the original. First A* will evaluate the accessible nodes to the ants then it chooses the best node to follow using a mathematical equation. Applying AntStar into a single-source shortest-path problem is efficient because of the improved algorithm.

The authors in [8] focus on the application of ant colony optimization. The tour planning is determined by the user's experience. If the tour planning, went well it can save a lot of time and operation costs. Tour planning depended on rules of thumb which resulted in wasted tourism costs, with the help of the Ant Colony optimization algorithm this problem will be solved.

Tour route planning based on the ant colony system article states that a random algorithm was used on the ant colony system, which is used to find the shortest path from A to B. The introducing algorithm will help ACO with calculating the distance using a dynamic planning ability which will balance the load on the nodes to make the algorithm effective [9].

Authors [10] in that TSP is a popular problem that is easy to formulate, difficult to solve, and has a lot of applications. TSP denotes that a salesman spends his time visiting n cities in cycles, which means in one tour he will visit all nodes and finishes at the starting node. ACO is the most efficient way to solve the TSP problem, but it has some limitations that can be solved using certain strategies.

ACO can help to solve robotic path planning problems. Ants make pheromones on the nodes but not arcs, which builds up a trail in a form of marks. ACO will tour around the nodes and give the best strategy to go through nodes. By creating a grid model for the robot navigation system ACO will be able to solve the robot path planning. The pheromone updating strategy has a major drawback which makes the algorithm have bad robustness [11].

Authors in [12] there are two main types of concurrencies, time concurrency, and information concurrency. Which depends on the stages that were performed in parallel, and the information shared

by them. Scientists have tried different methods to optimize the cycle time and cost but failed. ACO solved the problem which was successful with optimizing discrete. Sequential ordering problems for crane scheduling in port terminals states that sequential ordering problem is an ATSP that has constraints on vertices, a tour will be feasible only if these constraints are respected, which will help find a solution with less cost. This solution can be used in transportation or production planning. The algorithms that can be used to solve the problem come in different components such as LS, an ACS, and an HMT [5]. This paper [13] proposes a shrewd, ant-colony-based path planning calculation with built-in contemplations to maintain a strategic distance from steep and congested pathways through beautiful ranges. The algorithm was outlined to be natural and helpful in expansion to effectiveness; it gives quick reaction time without consuming excess framework assets, making it well-suited to mobile devices. Recreation comes about moreover appeared that the proposed algorithm outflanks the conventional subterranean ant colony calculation regarding merging speed and look range. Gathering grouping arranging (ASP) can essentially move forward get together exactness and decrease gathering costs in present-day fabricating businesses. Huge reflector antennas are troublesome to amass and direly require ASP. Based on genetic calculations (GAs) and ant colony optimization (ACO), an approach for ASP of reflector radio wires was created. Precise recreation of the get-together of the reflectors was required for the assessment and optimization of the ASP [14]. This research [15] includes a recommendation of an unused method to discover the ideal way for centralized and competitive multi-robot within the same energetic environment. The method utilized to crossbreed the pheromone path upgrading of MAX– MIN ACO (MMAS) calculation with D* calculation methodologies. Several experimental comes about mimicked distinctive dynamic environments for a diverse number of robots demonstrating that the proposed strategy performed well. The D* algorithm is a common algorithm that can be applied to deal with problems in robot motion planning.

The critical errand of replunging terrible or lost cells with sound seedlings is carried out by programmed transplanters. Two sorts of seedling plates, with 50 and 200 cells, were utilized. The ACA and the GA were found to have more focal points than the CSM in adding up to visit lengths for one plate. In addition, the ACA performed better than the GA when the numbers of purge cells and solid seedlings within the target and source trays expanded [16].

Authors in [17] have implemented The surface quality inspection of the industrial printed circuit board is a vitally important link in its manufacturing process. To inspect surface defects of PCBs effectively, the automatic optical inspection technology, in which the PCB image acquisition depends on the path planning method, is widely adopted by the industry. In this paper, by improving the ant colony algorithm, we devise a PCB image acquisition path planning model and the corresponding solving algorithms. Because the ACA encounters difficulty escaping from the local optimal solution, an improved ACA with a negative feedback mechanism is proposed that can obtain a better tour path with a higher probability. Finally, via simulation experiments, the proposed global negative feedback ACA can shorten the average length of the tour path by 1.7% without changing the time complexity.

Researchers in [18] did Tourism planning as a vital link in tourism. Compared with traditional tourism planning based on experience, it is more scientific and reasonable to formulate through mathematical modeling methods. This paper mainly studies the construction of a tourism planning information system based on the ant colony (AC) algorithm. In the solution process, for the problems with more attractions, you need to divide the area first, then solve each area separately, and then transform the result of the solution into the regional self-driving tour route planning, and finally form a self-driving tour route planning. The experiments in this article found that most of the area tour time is closer to 15 days, which reduces the number of outings in a year and effectively reduces the round-trip time. In this paper, the system construction of self-driving tour route planning problems and ideas for solving specific problems are suitable for route planning in scenic spots or scenic spots and have certain reference values.

Some Researchers in [8] they interesting metaheuristic techniques for defining the shortest path is the ant colony optimization algorithm. A considerable number of maps for the shortest path have been considered in time past using classical techniques which is appropriate for deterministic variables. This is possible by mimicking the path navigation and swarm propensities of natural entities to provide real-time quality geographical images representing diverse areas or terrain for easy access to routing and path planning for sustainability and economic benefits in systems. In this research, the solution power of ACO has been demonstrated to predict customers' behavior in a popular retail outlet, using the

traveling salesman problem for stochastic shortest path during the purchase of items in a big-box facility with multiple products and sixteen sections.

Some Authors in [19] want to obtain diverse good solutions, we propose a multi-population ant colony system algorithm. The experimental results show that the proposed algorithm can obtain diverse good solutions. Furthermore, the proposed algorithm is utilized to deal with a range of practical problems, which indicates that the proposed algorithm is of practical significance.

The Vehicle Steering Issue with Time Windows (VRPTW) may be a well-known combinatorial optimization problem found in numerous viable operations. In this paper [20], we present a new pheromone lattice initialization approach for Subterranean ant Colony Optimization (PACO) that employments data extricated from the issue occasion at hand. Our computational tests show that PI-PACO performs way better than PACO. Dismantling Possibility Data Graph (DFIG) is displayed first. Based on this chart, the issue of disassembly grouping arranging is changed over into an issue of the searching ideal way in this coordinated and weighted graph. Then, a subterranean ant colony optimization calculation is presented, integrated with a portion moving organizer in a 3D framework, to construct the DFIG, and look ideal solution(s). The visit of the ant colony represents a conceivable item dismantling arrangement. At last, by a case think about, the proposed strategy appears [21].

The authors of this paper [22], make an MDGVRP by maximizing pay and minimizing costs, time, and spread, and after that, that applies an IACO calculation that focuses on viably enlightening the issue. VRP is one of the broadly explored districts in transportation science, fundamentally due to the potential brought speculation reserves and advantage upgrade openings it brings to organizations. The calculations utilized in this examine livelihoods and creative approach in redesigning the pheromone that comes around in way better courses of action.

Yan Mei [23], The ideal way arranging for traveler guides in visitor ranges can viably progress the utilization of tourism time. This paper considers and analyzes the application and advancement of subterranean insect colony calculation in terminal tourism course arranging beneath the Android stage, and proposes an ideal visit direct way arranging to demonstrate. The exploratory comes about to appear that the proposed show and the optimized way arranging calculation are more optimized.

The authors of this paper [24], production system Sequencing involves the venture of selecting the nice method plan, that could decrease the time and value of producing a part, sub-meeting, or assembly. P-Seq is an extension of classical mission-allocation problem, wherein n duties are required to be assigned to m machines with the least operation price incurred for the entire undertaking. in this paper, we carried out a polynomial-time reduction if you want to rework an instance of P-Seg to GTSP with priority constraint GTSP-Ord. Such novel transformation allowed us to deal with the procedure sequencing problem as an NP-hard problem much like TSP.

The authors of this paper [25], Comprehensive route planning is the key technology in the design of unmanned surface vehicles. This paper establishes modeling of the global environment based on electronic graphics and hexagonal grids which are better than square grids in terms of validity, security, and speed. We introduce the Cube coordinate system to simplify hexagonal algorithms. In addition, we provide an improved algorithm to perform route planning between two points. This paper [26], proposes a singular BPPA by way of reusing the dispensed pheromones of the ACO approach. The results of the experiment reveal that BPPA explores new paths which might be suitable to detour small clusters of newly seemed boundaries. But, the incorporation of these excursions shrinks the authentic path and generates further gain.

A comparison of all the works is presented in tables 1 and 2 given below. The works are compared on various factors.

Material and Methods

Optimization Algorithms

One of the challenges we faced when we were just starting with our proposed solution is choosing which algorithms to choose from the sea of available algorithms that may or may not help our case. When we researched about the possible algorithms, we found out that we can split them into classes which are Heuristics, Meta-Heuristics, divide-and-conquer, dynamic programming and greedy algorithms (Note: This is by no means a comprehensive list of all the classes of algorithms) and the algorithm we chose was the Ant Colony Optimization which we did a comparative analysis against

a list of its counterparts:

- A* (pronounced A star)
- Djikstra algorithms
- Bellman-ford algorithm
- Johnson's algorithm
- Floyd-Warshall algorithm
- Greedy
- Christofides
- Simulated Annealing (SA)
- Threshold Accepting (TA)

Below is some brief introduction about the algorithms listed above and give some information about some needed information to read comfortably and to establish some understanding about the subject matter. Afterwards, we will delve deeper into the comparative analysis between them and against ACO.

Authors	Title	Application Area	Techniques	Modification to Algorithm	Tool Availability	Dataset Used	Geography	Performance Metrics	Results	Drawbacks
Xiao, Zheng and Wang, Zhenan and Liu, Deng and Wang, Hui.	A path planning algorithm for PCB surface quality automatic inspection	Path planning	ACA · Location adjustment	x	x	x	x	Time	The GNF-ACA can shorten the path length by 1.7% on average compared with the basic ACA	x
Qu, Z.	Construction of Tourism Planning Information System Based on Ant Colony Algorithm	Tourism Planning Information System (TPIS)	ACO	x	x	x	x	Results	The three methods of location adjustment can further shorten the length of the tour path by 5.6%, 13.1%, and 13.7%.	x
Okwu, M.O., Lagouge, T.K., Afenogho, J.O.	Application of ant colony optimizer (Aco) for effective path planning in a big-box store or retail facility	Big-box store or retail facility	ACO	x	x	x	x	Results	Obtained showed that there was a sharp decrease in the tour cost when it was applied to most traveling salesman problems.	x

Liu, H., Yu, Z., Zhang, W., Ma, Z.	Time-Limited Tour Planning Based on Ant Colony Optimization Algorithm	Time-Limited Tour Planning	ACO	Calculate the travel distance and memorize the optimal travel route when the route mutates with ω chance.	x	x	x	Time	The overall entertaining time cannot exceed 330 min.	x
Cheng, Y.-B., Huang, T., Huang, H.T., Gong, Y.-J., Zhang, J.	Multi-Population Ant Colony System for Multiple Path Planning of Food Delivery Applications	Food Delivery Applications	ACO	x	x	x	x	Time	ACS algorithm is an effective method to solve TSP.	x
Hasan, A.H., Majid Mosa, A.	Multi-Robot Path Planning Based on Max-Min Ant Colony Optimization and D∗ Algorithms in a Dynamic Environment	Multi-Robot Path Planning	ACO	x	x	x	x	Path	The results gotten from this algorithm are compared to those obtained from PSO- and Different Evolution DE by using two metrics Average Total Path Traversed (ATPT)and Average Uncovered Target Distance (AUTD	x
Chen, X.-Q., Hu, D.-W., Yang, Q.-Q., Hu, H., Gao, Y.	An improved ant colony algorithm for multi-objective vehicle routing problem with simultaneous pickup and delivery	Vehicle routing	IACO	x	x	x	x	Costs, time and spread	The equations used in this study take a unique approach to redesigning the pheromone so that it may be used in far better ways.	x

Mei, Y.	Study on the application and improvement of ant colony algorithm in terminal tour route planning under Android platform	Terminal tour route planning (Android)	ACO	x	x	x	x	Time	The model and algorithm's path planning outcomes are 315 more optimal, with shorter tour planning distances 316 and more compact tour scheduling.	x
Sarkar, A.	Application of ant colony system metaheuristic algorithm in manufacturing process sequencing problem	Plan selector (tourism)	ACO	x	x	x	x	Time, cost	Reduce the time and cost of making a part, a submeeting, or an assembly	x
Wang, Y., Yu, X., Liang, X.	Design and implementation of global path planning system for unmanned surface vehicle among multiple task points	Routing	A*	x	x	x	x	Distance, Time, Path found	Better route planning algorithm between two places	x
Meier, D., Tullumi, I., Stauffer, Y., Dornberger, R., Hanne, T.	A novel backup path planning approach with ACO	Path Planning	ACO	x	x	x	x	Distance	BPPA analyzes unique routes that might be used to detour small groups of recently discovered borders.	x
Zhang, W., Gong, X., Han, G., Zhao, Y.	An Improved Ant Colony Algorithm for Path Planning in One Scenic Area with Many Spots	Improving path planning	ACO	x	x	x	x	Results	Proposed algorithm outperforms the traditional ant colony algorithm in regards to convergence speed and search range.	x

Wang, D., Shao, X., Liu, H., Ge, X.	Assembly sequence planning for panels of reflector antenna based on hybrid algorithm	Improve assembly accuracy and reduce assembly costs	ACO	x	x	x	x	Time, distance	He computation speed of the GAACO algorithm is relatively faster than GA, and the algorithms of GA and GAACO have great stability. The combination of GA and ACO improves the convergence to an optimal sample.	x
Shi, W., Weise, T., Chiong, P.R.R., Catay, B.	Hybrid PACO with enhanced pheromone initialization for solving the vehicle routing problem with time windows	Using hybrid PACO to solve a problem	Hybrid PACO and ACA	ACA and hybrid PACO were used as thet are connected to ACO but modified	x	x	x	Results	The results of the ACA were better than those of the other two methods when the number of empty cells was relatively high.	x
Wang Hui, Niu Qiang, Xiang Dong, Duan Guanghong	Ant colony optimization for disassembly sequence planning	Optimizing disassembly sequence planning	ACO	x	x	x	x	Distance, Time, Path found	By a case study and experiments, at the aspect of searching for optimum solution(s), the proposed algorithm has a good and stable performance.	x
Deng, X., Zhang, L., Luo, L.	An improved ant colony optimization applied in robot path planning problem	Robot path planning	Ant colony system to find the best strategy	x	x	x	x	x	Find best iteration strategy and best nodes to go through	bad robustness

Huanwg, H.-C.	The application of ant colony optimization algorithm in tour route planning	Tour Route planing	Ant system	It was used to solve Traveling sales man problem	x	x	x	x	Shortest path from A to B	Stagnation phase
Tyagi, S.K., Yang, K., Verma, A.	Non-discrete ant colony optimisation (NdACO) to optimize the development cycle time and cost in overlapped product development	optimizing cycle time for product development	x	x	x	x	x	x	Reduce the cycle time	overlapping of more than two stages
Wang, P.-D., Tang, G.-Y., Li, Y., Yang, X.-X.	Improved ant colony algorithm for traveling salesman problems	salesman spends time visiting n cities in cycles	Constructing routes, phenomenon update, local search	x	x	x	x	x	Limit the number of candidates in a city	x
Yipeng, Z., Junping, D., Feng, X., Yang, Y., Xuyan, T.	Tour route planning based on ant colony system	Algorithm that improves ACO by calculating the distance	Dynamic planning	Used dynamic planning to improve the distance calculation	x	x	x	Distance	Find a better route with shorter distance with fewer tourists	x
Montemanni, R., Smith, D.H., Rizzoli, A.E., Gambardella, L.M.	Sequential ordering problems for crane scheduling in port terminals	Algorithm that helps with crane scheduling in ports	Find a feasible solution with minimum cost	Asymmetric Travelling salesman problem with precedence constraints	x	x	x	Time, Cost	Find a solution with minimum cost	x
Wu, Z., Zhang, L., Wang, Y., Wang, K.	Optimization for multi-resource allocation and leveling based on a Self-Adaptive ant colony algorithm	Schedule generation schemes	ACO	A self-adaptive variant of ACO	x	x	x	Time	The Self-adaptive variant was faster than the original implementation of the original approach	x

Montemanni, R., Smith, D.H., Gambardella, L.M.	A heuristic manipulation technique for the sequential ordering problem	Sequential ordering problems	ACO	Artificial Precedence Constraints	x	x	x	Results	Better Results when compared with the results of ACO without artificial precedence constraints	x
Montemanni, R., Rizzoli, A.E., Smith, D.H., Gambardella, L.M.	Sequential ordering problems for crane scheduling in port terminals	Maritime transport planning	ACO	ACS+LS+HMS (Ant Colony System + Local Search + Hueristic Manipulation Technique	x	x	x	Results	ACS+LS+HMS presented better computations than any other variant of ACO	x
Zbigniew ŚWiątnicki	Application of Ant Colony Optimization Algorithms for Transportation Problems Using the Example of the Travelling Salesman Problem	Transportation	ACO	x	x	x	x	results	Different heuristics and controlled parameters showed better results	x
Ramlakhan Singh Jadon, Unmukh Datta	Modified Ant Colony Optimization Algorithm with Uniform Mutation using Self-Adaptive Approach for Travelling Salesman Problem	Improving Route calculation for ACO	ACO	Uniform Mutations	x	x	x	Distance	The Variant returned better results than the original algorithm	x

Table a

A Very Brief introduction about graphs and graph theory

Graph theory is a branch of mathematics that studies graphs which is a structure used in math and software to model pairwise relations such as in our case places that are passed by the main processes to the algorithm to process and calculate the most optimal path. A place or an object is called a node or vertex and the connection between every pair of nodes is called an edge or a link.

There are many types and shapes that graphs come in. Directed vs Undirected, Space vs Dense, simple, connected, complete, cyclic, acyclic, bipartite, complete bipartite, and many more. Using any of them is highly dependent on the need and project utilizing them. In our case, we used a multiple edges complete graph. it was somewhat complex to implement but it's the nearest realistic graph type that we could think of to implement the roads system we have.

Every node and edge has attributes that we can make use of or just for storing information to be processed later. For example, in this project edges have an attribute called 'Weight' and it's responsible for storing the distances between each node and the other nodes around it. Inside the comparative analysis, we conducted the algorithms then are using this attribute to calculate the shortest path inside a particular graph.

An introduction to the algorithms

There are multiple classes of algorithms that are used to solve NP-complete problems we will only discuss three of them which are approximation, randomization, and heuristic algorithms. We will just touch on these four classes to give an idea to the reader on the different methods towards reaching a certain desired solution to a given problem.

Approximation algorithms are algorithms that are efficient in finding the approximate solution to an optimization problem they are different than the other classes in a way that it must come with a provable guarantee on how close the approximated solution is from the optimal solution. Randomized algorithms are algorithms in which randomness is integrated into the logic of their procedures. They typically use a set of random bits to guide its process to help it achieve a good performance.

Heuristics algorithms are algorithms that are developed to solve a very specific problem. They are faster and more efficient than the traditional and common algorithms used. Because everything comes with a price, it sacrifices accuracy, optimality. These algorithms are often used to solve NP-hard problems like the traveling salesman problem. There is another type of heuristic called meta-heuristic. They are not tailored to a specific problem like the way heuristics are.

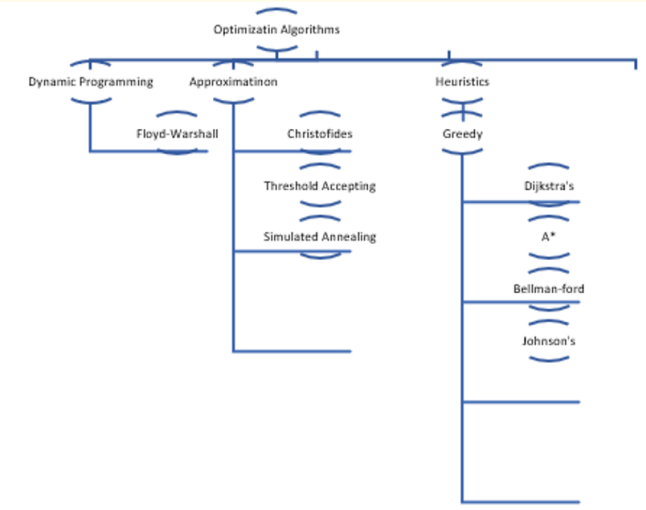

Figure a

Note: The pictures below are a set of simulations done on a graph that was passed into every single algorithm found below. We were testing the space and time consumption of each algorithm. The results displayed are not the most accurate ones because the current implementation of the algorithms listed below is not the most optimal in regards to the software aspect.

Introducing the ant colony optimization (ACO) algorithm

ACO was developed when behavioral scientists observed the foraging behavior of ants. The Analogy of using ants as an agent to explore a certain graph. when they observed real ants, they saw that ants use many techniques to leave a mark for other ants from their colony to follow them towards the food source and back to the colony. One of the many things that could be implemented in an algorithm is the pheromone trail they leave plus the attributes they carry with them, like evaporation rate or the destination it leads to. There are two types of pheromones that we will focus on one for a route to the food source and the other one is to the way back to the colony. When we go back to implementing this algorithm into the digital world. We found out if we scatter some ants throughout the graph, traversing the graph will become much faster compared to the other algorithms in this regard This type of system is called a multi-agent system. The algorithm itself is simply relative to the real-life counterpart that is based on. First, Scatter the ants throughout the graph. Second, Make the ants choose a random path to follow and they will drop the pheromones. In the digital implementation of the algorithm, we just add 1 to the weight that is set by default to 0 to the trail (edge) that the ants followed, Because the path with greater weight will be likely to be followed by more ants and be considered as the optimal option. Third, we then calculate the shortest path based on the total weight of the edges that the ants followed to return to their starting point and this is the last step in the ACO algorithm.

Dijkstra's algorithm

Dijkstra's algorithm is an algorithm developed to find the shortest path between nodes in a weighted graph. It is as fast as BFS (Breadth-first Search) faster only because it uses a priority queue rather than a common queue to store the visited nodes.

- This is the time complexity of Dijkstra's algorithm vs BFS $\Theta((|V| + |E|) \log |V|)$
- BFS's Of course, this may vary depending on the input graph $O(|V| + |E|) = O(b^d)$

A* Algorithm

A* search algorithm is an algorithm that is known for its completeness and optimality. One of its characteristics is that it is a best-first search method when it transverses a weighted graph. First, you need to specify the source node as a starting point then

```
In [15]: t = nx.dijkstra_path(g,'The Chedi Al Bait, Sharjah - a GHM hotel','Nation Towers')
         t

Out[15]: ['The Chedi Al Bait, Sharjah - a GHM hotel',
          'Palm Jumeirah',
          'Burj Al Arab',
          'Burj Khalifa',
          'Nation Towers']
```

Figure 1: Dijkstra's algorithm results.

it will advance till it finds the first-best solution. A* is an extension of Dijkstra's algorithm. It extended the mentioned algorithm by introducing a heuristic function into the calculations of Dijkstra's Algorithm.

$f(n) = g(n) + h(n)$, g(n) is the weight or cost of the starting node the next one and h(n) is the heuristic function used to estimate the lowest cost of the next possible node in the graph.

```
In [26]: t = nx.astar_path(g,'Emirates Palace', 'The Chedi Al Bait, Sharjah - a GHM hotel', weight="weight")
         t

Out[26]: ['Emirates Palace',
          'Nation Towers',
          'Burj Khalifa',
          'Burj Al Arab',
          'Palm Jumeirah',
          'The Chedi Al Bait, Sharjah - a GHM hotel']
```

Figure 2: A* Results.

Bellman-ford algorithm

While the Bellman-ford algorithm is slower than Dijkstra's algorithm it can calculate the shortest path concerning negative edge weights. Despite the differences between it and Dijkstra's, it works on the same principles of approximation called relaxation. In essence, both algorithms work by storing the current lowest cost solution than replacing it with a better solution during its run time. This gives it a better edge when compared to the other ones like A*.

```
In [14]: t = nx.bellman_ford_path(g,'The Chedi Al Bait, Sharjah - a GHM hotel','Nation Towers')
         t

Out[14]: ['The Chedi Al Bait, Sharjah - a GHM hotel',
          'Palm Jumeirah',
          'Burj Al Arab',
          'Burj Khalifa',
          'Nation Towers']
```

Figure 3: Bellman-Ford Results.

hnson's algorithm

This algorithm works by finding the shortest path and it integrates the Bellman-Fords algorithm into transforming the input graph into one with no negative weights then it is passed to Dijkstra's algorithm to be used to find the shortest paths. Because it uses a slight variant for implementing Dijkstra's the overall time complexity becomes faster than the Floyd-Warshall algorithm which I will introduce after this algorithm.

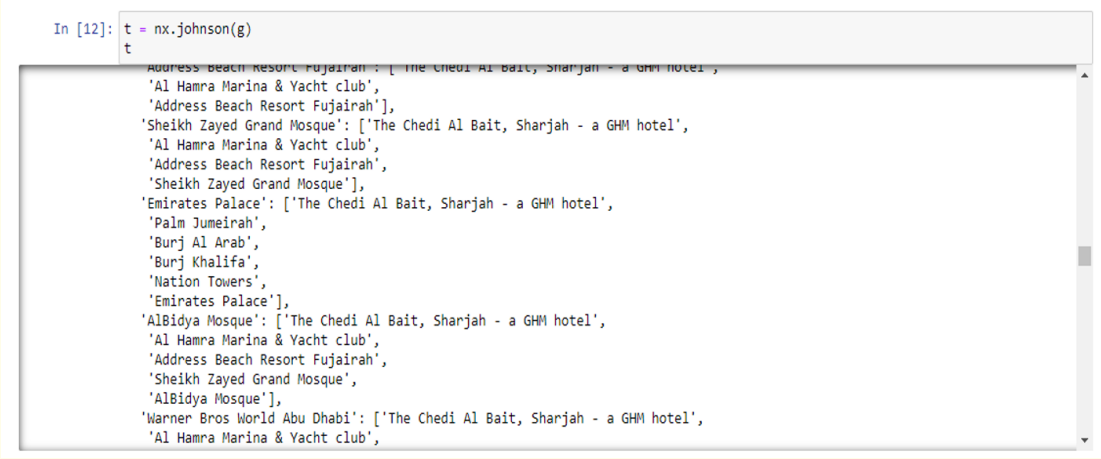

Figure 4: Johnson's algorithm results.

Floyd-Warshall algorithm

Floyd-Warshall algorithm is that finds the shortest path in a weighted directed graph between all pairs of vertices in it. Because of how the algorithm is developed when it compares the possible paths it $\Theta(|V|^3$ will do so with a which means every single vertex to be run inside a triple nested loop, Something like this.

```
for ( i = 0; i < N; i++ ) {
  for ( j = 0; j < N; j++ ){
       for ( h = 0; h < N; h++ )
    statement;
       }
}
```

Which may affect the growth of the graph with time.

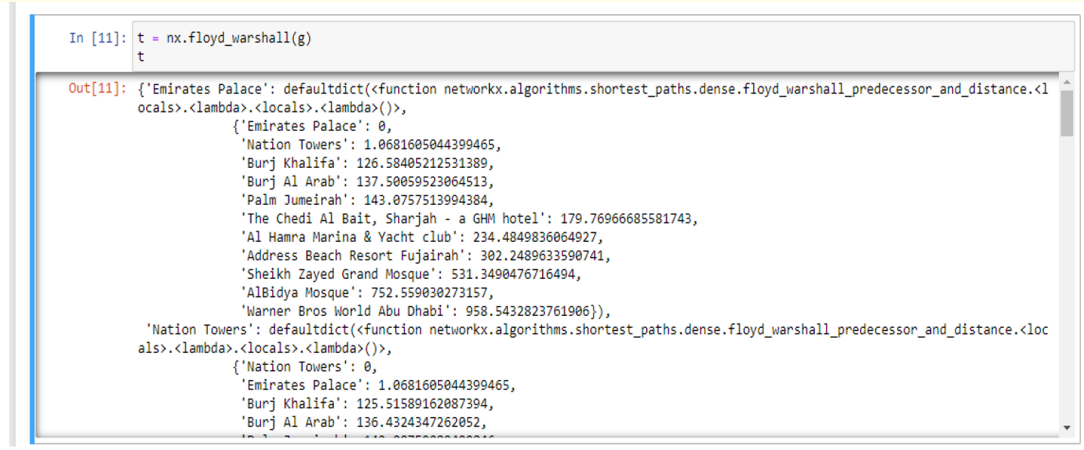

Figure 5: Floyd-Warshall results.

Greedy

Greedy algorithms are any algorithm that tries to find the locally optimal choice at each stage by following an admissible heuristic. Most of the algorithms used here do not return the most optimal path, because it is going to take an unreasonable amount of time and resources. However, by finding the locally optimal choice in each stage of the pathfinding we can merge them to find the nearest globally optimal solution. In our case, we only have to find the nearest optimal path from our current node. Dijkstra's and A* algorithms are such examples of Greedy algorithms.

```
In [18]: t = nx.approximation.greedy_tsp(completegraph)
         t

Out[18]: ['Emirates Palace',
          'Nation Towers',
          'Sheikh Zayed Grand Mosque',
          'AlBidya Mosque',
          'Warner Bros World Abu Dhabi',
          'Address Beach Resort Fujairah',
          'Al Hamra Marina & Yacht club',
          'Burj Al Arab',
          'Palm Jumeirah',
          'The Chedi Al Bait, Sharjah - a GHM hotel',
          'Burj Khalifa',
          'Emirates Palace']
```

Figure 6: Greedy algorithm results.

Christofides algorithm

This algorithm is designed to solve the Traveling salesman problem by approximating some solutions to the problem. It's an approximation algorithm that is proven to provide a result within 3/2 approximate factor of the solution of the optimal solution length. It needs a complete graph to work calculate the length properly.

```
In [16]: t = nx.approximation.christofides(completegraph)
         t

Out[16]: ['Emirates Palace',
          'Burj Khalifa',
          'Sheikh Zayed Grand Mosque',
          'AlBidya Mosque',
          'Warner Bros World Abu Dhabi',
          'Address Beach Resort Fujairah',
          'Al Hamra Marina & Yacht club',
          'Burj Al Arab',
          'Palm Jumeirah',
          'The Chedi Al Bait, Sharjah - a GHM hotel',
          'Nation Towers',
          'Emirates Palace']
```

Figure 7: Christofides Algorithm results.

Simulated annealing (SA)

This belongs to the randomized type of algorithms. It is a metaheuristic used to approximate the global optimum of large search spaces. It is made to find near-optimal solutions to the Traveling salesman problem. The technique used in this algorithm by decreasing the probability of accepting worse solutions while it is running. When other algorithms accept worse solutions, they waste their time when they compare them with the better solu-

tions found later during their runtime. This is avoided when using the SA algorithm because of Its inherent design.

```
In [23]: t = nx.approximation.simulated_annealing_tsp(completegraph, nx.approximation.greedy_tsp(completegraph))
         t

Out[23]: ['Emirates Palace',
          'Nation Towers',
          'Sheikh Zayed Grand Mosque',
          'AlBidya Mosque',
          'Warner Bros World Abu Dhabi',
          'Address Beach Resort Fujairah',
          'Al Hamra Marina & Yacht club',
          'Burj Al Arab',
          'Palm Jumeirah',
          'The Chedi Al Bait, Sharjah - a GHM hotel',
          'Burj Khalifa',
          'Emirates Palace']
```

Figure 8: Simulated Annealing Results.

Threshold accepting

Threshold Accepting searches the graph locally to find the most optimal solution. When it starts to when it finds the first feasible solution and from there it goes to randomly select one of the neighbor solutions and it only accepts it if it improves the main function. With each improvement done to the main function then the current level is set as the threshold to not get a worse result into the current solution search space.

```
In [24]: t = nx.approximation.threshold_accepting_tsp(completegraph, nx.approximation.greedy_tsp(completegraph))
         t

Out[24]: ['Emirates Palace',
          'Nation Towers',
          'Sheikh Zayed Grand Mosque',
          'AlBidya Mosque',
          'Warner Bros World Abu Dhabi',
          'Address Beach Resort Fujairah',
          'Al Hamra Marina & Yacht club',
          'Burj Al Arab',
          'Palm Jumeirah',
          'The Chedi Al Bait, Sharjah - a GHM hotel',
          'Burj Khalifa',
          'Emirates Palace']
```

Figure 9: Threshold Accepting Results.

Hardware and software resources

This section talks about the tools that we have used to build our project. We have used different kinds of tools to build a simple user interface so that users are satisfied with the system. The reason why the user interface is the most important part of the project is that it is the closest thing to the user. The user interface must be simple and easy for users to navigate through windows without needing a guide.

Pycharm

We have been using PyCharm, an integrated development environment for python. Pycharm is an open-source free tool for all developers. It can be used for a lot of things such as creating web applications and building web applications.

Tkinter

We also used Tkinter which is a framework that provides you with simple ways to create Graphical User Interface elements with different widgets which are found in the Tk toolkit in Tkinter. Tkinter has a lot of widgets such as Labels, Entry fields, buttons, and more. Tkinter is a beginner-friendly framework that helped us build the tour planning application.

Jupyter notebook

A jupyter notebook is also open-source software that is used to create visuals, documents, and other multimedia resources. In our project, we used Jupyter notebook for the visualization of the map to easily write the code and fix errors accordingly. We used the GMAPS package which is a google map package that allows you to create a responsive map visually and using the GMAPS package you can access different methods to for example add a marker based on the coordinates of the place and much more.

SQLite online IDE

In our project, we have a database that has different tables to store users and places information. Our project creates a .db file that is called information. db file which has the database information saved in it. We use an SQLite online IDE to access the file and run SQL queries to see all information stored in the database. SQLiteonline is a free online tool that can be used to open database files with the extension .db or to even create database files from the website and then export it.

Anaconda navigator

Anaconda Navigator is free open-source software that allows you to launch different kinds of applications such as Jupyter notebook, Datalore, pycharm, and much more tools. We used anaconda to run the Jupyter notebook and access it to create jupyter notebooks. Anaconda is used by developers because it is a simplified way to launch applications and manage conda packages without using command-line commands which can sometimes take time and effort.

Methodology

DFD

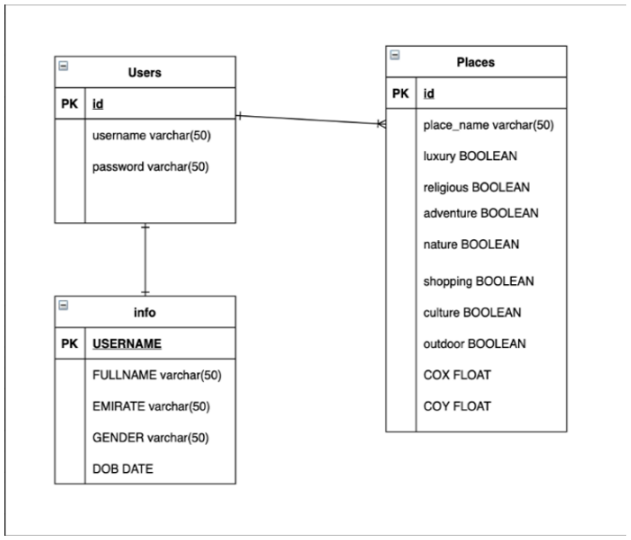

Figure b

Results

Dataset that been created

Simulation ResultsUnsorted

The results of the simulations done

Name	Total Time	Memory Used
ACO	0.210814 s	101.531 MiB
A*	0.0001687 s	101.137 MiB
Floyd_Warshall	0.00182 s	101.137 MiB
Johnson's	0.0019975 s	101.137 MiB
Bellman_Ford	0.000503 s	101.137 MiB
Dijkstra's	0.000118 s	101.199 MiB
Christofides	0.00906 s	101.262 MiB
Greedy	0.0002255 s	101.262 MiB
Simulated Annealing	0.0748691 s	101.266 MiB
Threshold Accepting	0.0728287 s	101.266 MiB

Table 2: Simulation Results Unsorted.

ID	Places	coX	coY
1	Emirates Palace	24.4617	54.317283
2	Nation Towers	24.4644	54.327411
3	Burj Khalifa	25.1973	55.274376
4	Burj Al Arab	25.1413	55.185263
5	Palm Jumeirah	25.1136	55.139121
6	The Chedi Al Bait, Sharjah - a GHM hotel	25.3593	55.383367
7	Al Hamra Marina and Yacht club	25.6974	55.780166
8	Address Beach Resort Fujairah	25.4886	56.414186
9	Sheikh Zayed Grand Mosque	24.4142	54.475662
10	AlBidya Mosque	25.440436	56.35444176

Table 3: The results of the simulations done.

The dataset visualized

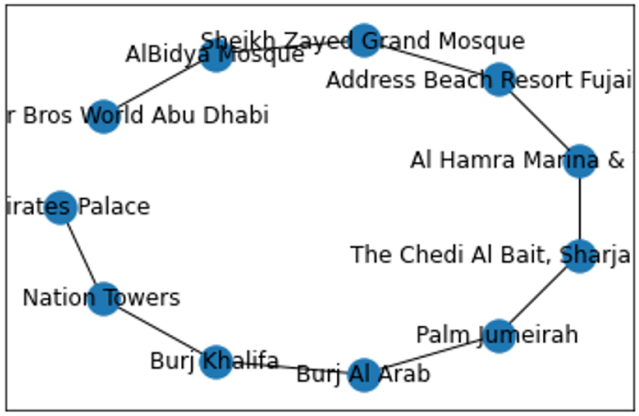

Figure 10: The Dataset visualized.

A Detailed Step by step guide of our approach will help you to understand the various processes we used to reach our current point. First, we collected some well-known places in the UAE as our dataset. There were no criteria to choose from we choose this set. Then we calculated the distance between them using the geopy package. The earlier step of calculating the distances was used for all the algorithms other than ACO, to be used as the weight to calculate the shortest route possible. After calculating the weights, a complete graph was created to ensure that the four algorithms, Christofides, Greedy, Simulated Annealing, and Threshold Accepting could work properly the other algorithms could work with simple graphs and don't require a complete graph. The results in table 5.1, are not to be taken as hard evidence that an algorithm is faster than the other because that is highly dependent on the implementation of the algorithm at the code level, choosing the right data structures is very crucial when doing any optimization to the algorithm inside the code.

Name	Total_Time	Memory Used
Dijkstra's	0.000118 s	101.199 MiB
Greedy	0.0002255 s	101.262 MiB
Floyd_Warshall	0.00182 s	101.137 MiB
Astar	0.0001687 s	101.137 MiB
Bellman_Ford	0.000503 s	101.137 MiB
Johnson's	0.0019975 s	101.137 MiB
Christofides	0.00906 s	101.262 MiB
Threshold Accepting	0.0728287 s	101.266 MiB
Simulated Annealing	0.0748691 s	101.266 MiB
ACO	0.210814 s	101.531 MiB

Table 4: The Comparative analysis results.

The comparative analysis

The comparative analysis results

When we observe table 4, we see that the memory consumption didn't vary too much between the algorithms. One of the reasons that they appear similar from the memory consumption part is that the dataset is very small to show the real differences between each algorithm and the rest. However, if we see the total time column it becomes more apparent that there are some differences in the performance in the time that they took to finish finding the shortest distance.

The Fastest one was Dijkstra's algorithm while ACO was the slowest. The results would vary if we used a bigger graph to analyze what is more interesting is that the Greedy algorithm came second to Dijkstra's. If the Dataset was larger, The slowest four will have a chance to display their advantage in processing faster when given larger data to work with.

From the analysis done we found out that the original ACO will not serve our cause as intended. Because the original algorithm

can't specify the source node (starting point of the tour) which is very important in our case. Adding artificial precedence constraints into ACO will resolve the issue stated earlier and further improve the performance.

To conclude this part of the report. We would like to state that if you're interested in further exploring the algorithms above to implement them depending on reliable sources and software packages that are available online to ensure the best possible outcomes and results and to improve the results shown above we will strive to further deepen our understanding of problems and knowledge at hand.

Conclusion

In conclusion, we have started the application purpose and what target we are aiming for our application to take place in, stated the functionality, discussed the advantages of using the application, and the compilers we used together with the languages we mastered to write the code, APIs, algorithms, literature review, and the risk assessment. We hope that the project we are creating will reach the world someday and be used by most people.

Bibliography

1. https://www.macrotrends.net/countries/ARE/uae/tourism-statistics

2. Z Wu., *et al*. "Optimization for multi-resource allocation and leveling based on a self-adaptive ant colony algorithm". in 2008 International Conference on Computational Intelligence and Security 1 (2008): 47-51.

3. Ramlakhan Singh Jadon., *et al*. "Modified Ant Colony Optimization Algorithm with Uniform Mutation using Self-Adaptive Approach" (2013).

4. Zbigniew Swiatnicki. "Application of ant colony optimization algorithms for transportation problems using the example of the traveling salesman problem" (2015).

5. R Montemanni., *et al*. "Sequential ordering problems for crane scheduling in port terminals". *International Journal of Simulation and Process Modelling* 5.4 (2009): 348-361.

6. R Montemanni., *et al*. "A heuristic manipulation technique for the sequential ordering problem". *Computers and Operations Research* 35.12 (2008): 3931-3944.

7. Mohammed Faisal., *et al*. "AntStar: enhancing optimization problems by integrating an Ant System and A* algorithm". *Scientific Programming* (2016).

8. HC Huang. "The application of ant colony optimization algorithm in tour route planning". *Journal of Theoretical and Applied Information Technology* 52.3 (2013): 343-347.

9. Y Zhou., *et al*. "Tour Route Planning Based on Ant Colony System". in 2009 WRI Global Congress on Intelligent Systems 1 (2009): 218-221.

10. Rongwei Gan., *et al*. "Improved Ant Colony Optimization Algorithm for the Traveling Salesman Problem". *Journal of Systems Engineering* 21.2 (2010).

11. X Deng., *et al*. "An Improved Ant Colony Optimization Applied in Robot Path Planning Problem". *Journal of Computational* 8.3 (2013): 585-593.

12. S K Tyagi., *et al*. "Non-discrete ant colony optimisation (Nd-ACO) to optimise the development cycle time and cost in overlapped product development". *International Journal of Production Research* 51.2 (2013): 346-361.

13. "Retrieving data. Wait a few seconds and try to cut or copy again".

14. D Wang., *et al*. "Assembly sequence planning for panels of reflector antenna based on hybrid algorithm". *Computer Integrated Manufacturing Systems* 23.6 (2017): 1243-1252.

15. A H Hasan. "Robot path planning based on improved max-min ant colony optimization algorithm in dynamic environment". *Research Journal of Applied Sciences* 11.10 (2016): 1060-1068.

16. Z Jiang., *et al*. "Comparing an ant colony algorithm with a genetic algorithm for replugging tour planning of seedling transplanter". *Computers and Electronics in Agriculture* 113 (2015): 225-233.

17. Z Xiao., *et al*. "A path planning algorithm for PCB surface quality automatic inspection". *Journal of Intelligent Manufacturing* (2021): 1-13.

18. Z Qu. "Construction of Tourism Planning Information System Based on Ant Colony Algorithm". in Journal of Physics: Conference Series 1533 (2020): 022101.

19. YB Cheng., *et al*. "Multi-population ant colony system for multiple path planning of food delivery applications". in 2018 IEEE Symposium Series on Computational Intelligence (SSCI) (2018): 68-73.

20. W Shi., *et al*. "Hybrid PACO with enhanced pheromone initialization for solving the vehicle routing problem with time windows". in 2015 IEEE Symposium Series on Computational Intelligence (2015): 1735-1742.

21. "Ant colony optimization algorithms – Wikipedia" (2021).

22. Y Wang., *et al*. "An improved ant colony optimization algorithm to the periodic vehicle routing problem with time window and service choice". *Swarm and Evolutionary Computation* 55 (2020): 100675.

23. Y Mei. "Study on the application and improvement of ant colony algorithm in terminal tour route planning under Android platform". *Journal of Intelligent and Fuzzy Systems* 35.3 (2018): 2761-2768.

24. A Sarkar and D Sormaz. "Application of Ant Colony System Meta-heuristic Algorithm in Manufacturing Process Sequencing Problem". in IIE Annual Conference. Proceedings (2018): 1474-1479.

25. Y Wang., *et al*. "Design and implementation of global path planning system for unmanned surface vehicle among multiple task points". *International Journal of Vehicle Autonomous Systems (IJVAS)* 14.1 (2018): 82-105.

26. D Meier., *et al*. "A novel backup path planning approach with ACO". In 2017 5th International Symposium on Computational and Business Intelligence (ISCBI) (2017): 50-56.

27. https://stackoverflow.com/questions/4539542/sqlite-vs-sql-server

28. https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-manag